

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

REMARKS

Claims 1 to 21 were pending in the Application at the time of examination. Examiner has rejected Claims 1 to 21 under 35 U.S.C. § 103(a) as obvious over the Dukach et al. reference (US 6,609,159 B1) in view of the Woodring et al. reference (US 6,519,686 B2) and in further view of Jason Lango ("An implementation of the Solaris Doors API for Linux,") a non-patent literature reference dated 1998. Claims 1 to 21 remain in the application as originally filed.

REJECTION OF CLAIMS 1 - 21 UNDER 35 U.S.C. 103(a)

The Examiner rejected Claims 1 to 21 under 35 U.S.C. § 103(a) as obvious over the Dukach et al. reference (US 6,609,159 B1) in view of the Woodring et al. reference (US 6,519,686 B2) and in further view of "An Implementation of the Solaris Doors API for Linux," by Jason Lango.

Applicants respectfully traverse Examiner's rejection of Claims 1 - 21 based on obviousness considerations for the reasons that follow herein.

Applicants first respectfully submit that their reply to Examiner Office Actions of June 13, 2005, and March 08, 2007, have already overcome the obviousness rejections of Dukach et al. in view of Woodring et al.

Applicants Independent Claim 1 reads as follows with emphasis added:

A method for moving data between processes in a computer-based system, each process calling for one or more symbols in a first library; the method comprising:

associating each process with a second library, said second library comprising one or more symbols with a **door interprocess communication mechanism**, said door interprocess mechanism enabling each process to set up an initial connection, said connection subsequently communicating a synchronization signal using a semaphore, said one or

GUNNISON, McKAY &
RODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Menlo Park, CA 94025
(811) 655-0880
Fax (811) 655-0888

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

more symbols enabling data communication through a mapped memory based on said synchronization signal; intercepting a call from each process for a symbol in said first library; and redirecting said call to a corresponding symbol in said second library.

Applicants Independent Claim 7 reads as follows with emphasis added:

A program storage device readable by a machine, tangibly embodying a program of instructions readable by the machine to perform a method for moving data between processes in a computer-based system, each process calling for one or more symbols in a first library, the method comprising:

associating each process with a second library, said second library comprising one or more symbols with a **door interprocess communication mechanism**, said door interprocess mechanism enabling each process to set up an initial connection, said connection subsequently communicating a synchronization signal using a semaphore, said one or more symbols enabling data communication through a mapped memory based on said synchronization signal; intercepting a call from each process for a symbol in said first library; and redirecting said call to a corresponding symbol in said second library.

Applicants Independent Claim 13 reads as follows with emphasis added:

An apparatus for moving data between processes in a computer-based system, the apparatus comprising:

a plurality of processes;
a mapped memory;
a first library having one or more symbols, said plurality of processes calling for said one or more symbols in said first library of symbols;
a second library having one or more symbols, said one or more symbols associated with a semaphore and a **door interprocess communication mechanism** setting up an initial connection; and

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

an interposer intercepting a call from a process for said one or more symbols in said first library and redirecting a call for corresponding said one or more symbols in said second library.

Applicants Independent Claim 21 reads as follows with emphasis added:

An apparatus for moving data between processes in a computer-based system, each process calling for one or more symbols in a first library, the apparatus comprising:

means for associating each process with a second library, said second library comprising one or more symbols with a **door interprocess communication mechanism**, said door interprocess mechanism enabling each process to set up an initial connection, said connection subsequently communicating a synchronization signal using a semaphore, said one or more symbols enabling data communication through a mapped memory based on said synchronization signal;

means for intercepting a call from each process for a symbol in said first library; and

means for redirecting said call to a corresponding symbol in said second library.

As shown above, each of Applicants independent Claims 1, 7, 13 and 21 includes the recited feature of a door interprocess communication mechanism, said door interprocess mechanism enabling each process to communicate a synchronization signal and/or an interposer/interposition, or equivalent wording to that effect.

The Examiner has again stated, (emphasis added):

Dukack teaches the invention substantially as claimed including: data (information, col 3, ln 35-42), data between processes in a computer-based system (col 6, ln 35-40/col 8, ln 37-42), one or more symbols (OS function 144, col 8, ln 55-62), the first library (the library of the OS 134, col 8 ln 52-55), process calling for one or more symbols in a first library (col 8, ln 58-62), associating each process

GUNNISON, McKAY &
RODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

with a second library (col 8, ln 36-37), a second library (the interposed library, col 8, ln 36-37/ln 60-65), one or more symbols of the second library (the interposed library function col 8, ln 52-65), **a door interprocess communication (file descriptor, col 3, ln 62-64/col 10, ln 33-34/ln 53-55), said door interprocess mechanism enabling each process to communication (col 15, ln 1-6/col 16, ln 15-21), interprocess communication mechanism (interprocess communication links, col 8, ln 40-46),** intercepting a call from each process for a symbol in said first library (col 8, ln 58-65/col 9, ln 24-30), redirecting said call to a corresponding symbol in said second library (col 8, ln 63-65).

Applicants again note that Dukach's col 8, lns 35-65 reads as follows (emphasis added):

The back end server and the interposed library which is linked to it, are one process. The front end server is another. The OS accords each separate process its own separate subspace within the common OS space. **A given process cannot directly write to another process's sub-space, but the OS does let it communicate with another processes in the same OS space through interprocess communication links, or pipes. Such pipes are defined and only work within a given OS space defined by a given OS kernel.**

As previously submitted, Dukach et al., only teaches or suggests the use of **pipes** for interprocess communications (IPC). Pipes are unidirectional and only allow processes of common ancestry to pass information such that an output of one process can be used as an input to another process. The above arrangement is clearly shown in FIG. 10 of Dukach et al. Moreover, the limitations associated with the use of pipe IPC are largely immaterial to Dukach et al., as the pipes established between the back end server 110 and front end server 112 (FIG. 10) are not impacted by the unidirectional nature, common ancestry or operating kernel limitations.

GUNNISON, McKAY &
RODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Menlo Park, CA 94025
(811) 655-0380
Fax (811) 655-0389

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

The use of pipe IPC simply allows for a limited set of networking calls to be intercepted by interposer library 116 and redirected by a pipe to the front end server 112 for processing, thus reducing the networking overhead of the back end server 110.

The main goal of Dukach et al. is to reduce the networking overhead of back end servers by rerouting a limited set of networking calls to the front end server using existing programming techniques which do not require rewriting of networking software or operating systems and thus maintains compatibility with existing software. This goal is clearly identified in Dukach et al. at Col 6, lns 42-49 as follows (emphasis added):

The interposed library returns from each back end call it intercepts with information in the same format as if the back end call had been executed directly by the OS, without the intervention of the library or front end server. This enables the front end 112 to insert the functionality provided by the programming 114 between the back end 110 and its clients 106 **without requiring any change to the back end server, other than its linking to the interposed library 116.**

However, in order to implement Solaris Doors, significant recoding of existing networking software and/or operating system of the front-end and back-end servers would be required as is stated on Page 5, lns 3-11 of Applicants' Specification as follows (emphasis added):

The fastest form of IPC on Solaris™ Operating System from Sun Microsystems Inc. is *doors*. However, applications that want to communicate using *doors* **need to be explicitly programmed to do so.**

GUNNISON, McKay &
RODGSON, L.L.P.
Garden West Office Plaza
1920 Garden Road, Suite 220
Menlo Park, CA 94025
(415) 655-0880
Fax (415) 655-0888

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

Accordingly, Dukach et al. effectively teaches away from using Solaris Doors as an IPC mechanism at least due to the labor and cost intensive process of recoding of existing networking and/or operating system software of the front-end and back-end servers.

Pipes, such as those specifically disclosed and taught in Dukach et al., are discussed in the "BACKGROUND OF THE INVENTION SECTION" of Applicants Specification at, for example page 2, ln 18 to page 3, ln 7. Pipes, such as those specifically disclosed and taught in Dukach et al., are also shown in Applicants FIG.1, clearly marked as "Prior Art". Page 2, ln 18 to page 3, ln 7 of Applicants Specification reads as follows, with emphasis added:

Interprocess communication (IPC) is the exchange of data between two or more processes. **Various forms of IPC exists: pipes, sockets, shared memory, message queues, and Solaris™ doors.**

A pipe provides the ability for a byte of data to flow in one direction and is used between processes. These two processes must be of common ancestry. Typically, a pipe is used to communicate between two processes such that the output of one process becomes the input of another process. FIG. 1 illustrates a conventional pipe 100 according to a prior art. The output of process 102 becomes the input of process 104. Pipe 100 is terminated when process 102 that is referencing it terminates. Data is moved from process 102 to process 104 through a pipe 100 situated within a kernel 106.

As shown above, **Applicants clearly distinguish pipes as distinct from doors** and then explain some of the limitations of pipes. As noted above, the Examiner further states that Dukach teaches (emphasis added):

a door interprocess communication (file descriptor, col 3, ln 62-64/col 10, ln 33-34/ln 53-

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

55), said door interprocess mechanism enabling each process to communication (col 15, ln1-6/col 16, ln 15-21)

Page 5, ln 3-11 of Applicants' Specification reads as follows:

The fastest form of IPC on Solaris™ Operating System from Sun Microsystems Inc. is doors. However, applications that want to communicate using doors need to be explicitly programmed to do so. Even though doors IPC is very fast, the socket-based IPC is more popular since it is portable, flexible, and can be used to communicate across a network.

A definite need exists for a fast IPC technology that would overcome the drawbacks of doors and socket-based IPC. **Specifically, a need exists for a fast socket technology implementation using doors.** A primary purpose of the present invention is to solve these needs and provide further, related advantages.

In light of the discussion above, Applicants respectfully submit that, contrary to the Examiners' comments, the disclosure of a "file descriptor" in the Dukach et al. reference does not teach, suggest or otherwise provide any motivation for the Solaris "doors" recited in Applicants Claims 1, 7, 13 and 21.

The Examiner has again stated, (emphasis added):

Dukack does not explicit teach a synchronization signal, a mapped memory. However, Woodring teaches a synchronization signal, a mapped memory, semaphore (the synchronization, management, and processing of the information stream, col 6, ln 20-22/ memory mapped file, col 7, ln 30-31/ map the buffer pointer to the buffer into their address space, col 7, ln 13-16/ semaphore (FBSEM) mechanism 316, col 6, ln 24-28). **It would have been obvious to one of the ordinary skill in the art at the time the invention was made to modify the teaching of Dukack and Woodring because Woodring's a synchronization**

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Menlo Park, CA 94025
(650) 653-0880
FAX (650) 655-0558

Page 13 of 18

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

signal, a mapped memory, semaphore would improve the efficiency of Dukack's system by providing an efficient information streaming of multimedia information in a multi-process software environment.

Applicants respectfully disagree with Examiner's determination for the reasons that follow. Woodring et al., is directed toward solving the problem of handling data intensive transport of streaming media to remote clients without having to clone multiple copies of the source media resident on a server for broadcast delivery to the remote clients; as such, Woodring has little to do with improvements to IPC techniques other than ensuring that the server and client are properly synchronized for broadcasting of the streaming media. Moreover, the arrangement of Woodring et al. is a well known version of shared memory IPC disclosed in Applicants specification.

Shared memory, such as those specifically disclosed and taught in Dukach et al., are discussed in the "BACKGROUND OF THE INVENTION SECTION" of Applicants Specification at, for example page 4, lns 1-22 to page 5, ln 1-2. Shared memory, such as those specifically disclosed and taught in Woodring et al., are also shown in Applicants FIG.3, clearly marked as "Prior Art". Applicants specification Page 4, lns 1-6 and is provided below.

Shared memory is another form of IPC. FIG. 3 illustrates the use of a shared memory 300 to communicate process 302 with process 304. Shared memory is an IPC technique that provides a shared data space that is accessed by multiple computer processes and may be used in combination with semaphores. Shared memory allows multiple processes to share virtual memory space. Shared memory provides a quick but sometimes complex method for processes to communicate with one another.

Since, Dukach et al. is under the control of a given OS kernel, there would be no need to provide semaphores for

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Menlo Park, CA 94025
(811) 655-0889
Fax (811) 655-0888

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

synchronization between processes as this is a normal operating system function provided with pipe IPC. Moreover, as discussed above, pipe IPC is unidirectional and only works with processes having a common ancestry within a given operating space. As such, the shared memory arrangement of Woodring et al. would not work with multiple unrelated processes within multiple operating system spaces as is shown in FIG. 4 of Woodring et al. Even if the two IPC techniques could be combined, the end result would add processing overhead for at least the back end server thus defeating the use of the interposed library. Moreover, since a stated goal of Dukach et al. is not **"requiring any change to the back end server, other than its linking to the interposed library 116,"** combining, if even possible, the shared memory IPC technique of Woodring et al. with the pipe IPC technique of Dukach et al. would require extensive recoding of at least the operating system software thus further defeating the goal of minimizing software changes. Col 6, lns 48-49 (Emphasis added.)

The Examiner further states, (emphasis added):

Dukack and Woodring do not explicitly teach door is a file descriptor/calls the OS read () function with that file descriptor. However, Jason teaches door is a file descriptor (Door are made visible to the application programmer as standard Unix file descriptor (or "door descriptor, sec: 4, In 8-10/ the door is represented in the kernel code as a door structure, sec: 6.3, In 3-6/ a door library call refer to a particular door via a door descriptor, the door kernel code will look in the current process's file descriptor table, sec: 6.1, In 13-16). **It would have been obvious to one of the ordinary skill in the art at the time the invention was made to modify the teaching of Dukack, Woodring with Jason to incorporate the feature of door is a file descriptor because this affects a big performance improvement over the pipe implementation.**

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Menlo Park, CA 94025
(811) 655-0880
Fax (811) 655-0888

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

Again, Applicant's respectfully disagree with Examiner's determination for the reasons that follow below. The new NPL reference by Jason Lango, "Lango", expressly requires recoding of existing applications in order to take advantage of doors for IPC on Linux based operating systems. In a first instance, the very title of the reference "An Implementation of the Solaris Doors API for Linux," is indicative of recoding of existing applications as the newly provided API (Application Programming Interface) requires that existing (and new) applications be specifically written to implement the Solaris Doors IPC as the set of routines used by the appropriately coded applications are what directs the operating system (e.g., Linux) to establish door IPCs. Lango expressly states in several locations that programming is necessary to implement the Solaris Doors API (emphasis added):

Doors are made visible to the **application programmer** as standard UNIX file descriptors (or "door descriptors"); at page 3, lns 3-4;

This allows the **programmer** the ability to control how many threads are actually created in the server and the initial parameters of the server threads (e.g., scheduling parameters, signal dispositions, etc.) at page 4, lns 16-19;

Since the **programmer** doesn't necessarily provide the memory block to receive arguments to and from door_call(3), the kernel may optimize the transfer of large arguments by mapping the underlying pages of memory into target process and copying a page only if a process attempts to modify it.; at page 4, lns 29-33.

The Solaris Doors API is an interesting new IPC mechanism. Its chief advantages are in the optimizations which can be made in the kernel implementation and how it automates the job of thread creation (in the general case) for the **server programmer**. at page 14, lns 2-4.

Accordingly, Applicants respectfully submit that the pipe IPC mechanism Dukach et al. in view of the shared memory IPC of

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1500 Garden Road, Suite 220
Monterey, CA 93940
(831) 633-0900
Fax (831) 633-0888

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

Woodring et al. and in further view of Solaris Doors IPC of Lango are rendered incompatible with the expressly stated goal of Dukach et al. of not **"requiring any change to the back end server, other than its linking to the interposed library 116."** Col 6, lns 48-49 (Emphasis added.) Applicants further respectfully submit that the reference to Lango is nothing more than a statement of prior art, which attempts to port over a limited implementation of a well known IPC technique to a more limited implementation of the UNIX operating system, Linux.

Accordingly, Applicants have demonstrated that the teachings of Woodring et al., are incompatible with the teachings of Dukach et al., and thus no *prima facie* case of obviousness is possible when viewed in light of one having ordinary skill in that art at the time of the invention.

Even under the broader standards promulgated by KSR, "it remains legally insufficient to conclude that a claim is obvious just because each feature of a claim can be independently shown in the cited art." (KSR Opinion at p. 14)

In the spirit of compact prosecution as articulated in MPEP § 707.07(g) to prevent unnecessary and wasteful piecemeal examination, Applicant respectfully requests reconsideration and withdrawal of Examiner's obviousness rejections in this case.

In light of the discussion above, Applicants respectfully request the Examiner to withdraw the rejection of Independent Claims 1, 7, 13 and 21 under 35 U.S.C. § 103(a) and allow Claims 1, 7, 13 and 21 to issue.

Since Dependent Claims 2-6, 8-12, 14-20 depend directly or indirectly from their respective Independent Claims 1, 7, 13, 21, inherit all of the features therefrom, and include additional features not found in Independent Claims 1, 7, 13, 21, Applicants respectfully request the Examiner to withdraw the rejection of Dependent Claims 2-6, 8-12, 14-20 under 35

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Menlo Park, CA 94025
(650) 655-0380
Fax (650) 655-0888

Appl. No. 09/991,596
Amdt. dated January 18, 2008
Reply to Office Action of 10/19/2008

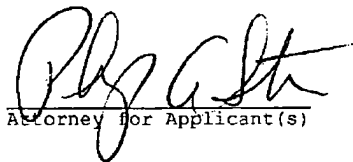
U.S.C. § 103(a) and allow Dependent Claims 2-6, 8-12, 14-20 to issue.

CONCLUSION

For the foregoing reasons, Applicants respectfully request allowance of all pending Claims 1- 21. If the Examiner has any questions relating to the above, the Examiner is respectfully requested to telephone the undersigned Attorney for Applicants.

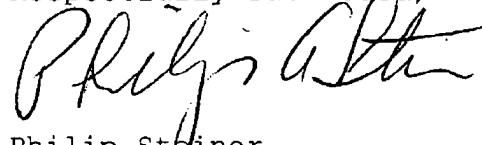
CERTIFICATE OF MAILING

I hereby certify that this correspondence is being transmitted by electronic means to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on January 18, 2008.


Attorney for Applicant(s)

January 18, 2008
Date of Signature

Respectfully submitted,



Philip Steiner
Attorney for Applicants
Reg. No. 47,967
Tel.: (831) 655-0880 x11